Green University of Bangladesh

Department of Computer Science and Engineering (CSE)
Semester: (Summer, Year: 2025), B.Sc. in CSE (Day)

Sarcasm Detection with Contextual Analysis Using Bi-LSTM

Course Title: Machine Learning Lab
Course Code: CSE-412
Section: 221-D2

Students Details

Name	ID
Obydullah Sayed	221902299
MD. Samirul Islam	221902253
Shabrina Sultana	221902321

Submission Date: 21-08-2025
Course Teacher's Name: Md. Riad Hassan

[For teachers use only: Don't write anything inside this box]

Assignment Status	
Marks:	**Signature:**
Comments:	**Date:**

Contents

1 Introduction 2
2 Background and Literature Review 2
2.1 Early Approaches to Sarcasm Detection 2
2.2 Machine Learning Era 3
2.3 Rise of Deep Learning 3
2.4 Contextual and Multimodal Approaches 3
2.5 Challenges and Gaps 3
3 Methodology 4
3.1 Dataset Creation 4
3.2 Data Preprocessing 4
3.3 Model Development 4
3.4 Evaluation Metrics 5
3.5 Hyperparameter Tuning 5
3.6 Computational Environment 5
4 Implementation 5
5 Results and Analysis 6
5.1 Bi-LSTM Training Results 6
5.2 Performance Comparison 7
5.3 Accuracy and Validation Curves 7
5.4 Analysis 8
6 Discussion 8
7 Conclusion 9

Abstract

Sarcasm detection is a critical yet challenging task in natural language processing (NLP) because it relies on understanding context, which is often missing in short texts like headlines or tweets. This report provides an extensive overview of sarcasm detection research, covering traditional methods, machine learning, and deep learning approaches, including recent models like Bi-LSTM, BERT, and RoBERTa. It also details a specific project that enhances sarcasm detection by creating a new dataset with full news articles and testing it with Bi-LSTM models. Starting with the `Sarcasm_Headlines_Dataset` (28,000 entries), we fetched 12,000 articles, filtered to 5,000 rows, and achieved accuracies of 81.84% (headlines only) and 95.44% (headlines + full articles). The report discusses broader trends, critiques limitations, and proposes future directions, including advanced models and larger datasets.

1 Introduction

Sarcasm is a way of using words to mean the opposite of what is said, often to mock or criticize. For instance, saying "Fantastic job!" after a mistake can be sarcastic if the intent is to point out failure. This subtle form of expression is common in everyday speech, social media, and news, making it important to detect for understanding true sentiments. However, sarcasm is tricky because it depends on context—details like the situation, tone, or surrounding text—that short messages often lack. Datasets like the `Sarcasm_Headlines_Dataset` or Twitter posts provide only headlines or brief posts, limiting a model's ability to grasp the full meaning.

This report explores sarcasm detection broadly, reviewing past and present research, and includes a specific study based on our work. We created a dataset with full article context and used Bi-LSTM models to test if this improves detection. Our results show a big jump in accuracy, from 77.84% to 95.44%, when context is added. The goal is to understand how sarcasm detection has evolved, highlight our contribution, and suggest future improvements. This work could help improve tools for sentiment analysis, fake news detection, and online safety, where missing sarcasm can lead to errors.

2 Background and Literature Review

2.1 Early Approaches to Sarcasm Detection

Early sarcasm detection focused on simple rules and patterns. Researchers looked at word choices, punctuation (like exclamation marks), or phrases that hint at sarcasm, such as "yeah, right." These methods worked for clear cases but struggled with subtle sarcasm because they did not consider context. For example, a phrase like "Great service!" could be sarcastic or genuine depending on the situation, and rule-based systems could not tell the difference. This limitation pushed the field toward more advanced techniques.

2.2 Machine Learning Era

Machine learning brought a shift by using features like word frequency, sentiment scores, and user behavior. Models like Support Vector Machines (SVM) and Naive Bayes were trained on labeled datasets, such as Twitter posts, to spot sarcasm. These approaches improved accuracy to around 70–85% on some datasets by combining lexical cues (e.g., positive words with negative intent) and statistical patterns. However, they still relied heavily on hand-crafted features, which were time-consuming and missed deeper context, like the conversation around a tweet.

2.3 Rise of Deep Learning

Deep learning marked a big step forward. Models like Long Short-Term Memory (LSTM) and Bidirectional LSTM (Bi-LSTM) can process text sequences, looking at words both forward and backward to capture context. Studies have shown Bi-LSTM achieving accuracies up to 93–98% on balanced datasets by learning semantic relationships. For instance, a hybrid model combining Bi-LSTM with attention mechanisms reached 97.87% accuracy on Twitter data by focusing on key sarcastic words.

Transformers, like BERT and RoBERTa, took this further by using pre-trained language understanding. BERT, which considers both sides of a sentence, has hit 98.5% accuracy on some datasets by fine-tuning on sarcasm-specific tasks. RoBERTa, an optimized version, improved performance by adjusting training methods, reaching 76.80% on Reddit data. These models excel with context but need large datasets and computing power, which can be a barrier.

2.4 Contextual and Multimodal Approaches

Recent research emphasizes context beyond single sentences. Models now use previous tweets, user profiles, or even images to detect sarcasm. A study using BERT with contextual cues achieved a 99% F1 score on news headlines by including surrounding text. Multimodal methods, combining text and images, have also emerged, with accuracies like 92.80% on Twitter datasets by fusing LSTM and BERT outputs. These approaches show that sarcasm often needs more than words—it relies on the full picture.

2.5 Challenges and Gaps

Despite progress, challenges remain. Small or unbalanced datasets limit model training, and low-resource languages (e.g., Marathi, Bengali) lack enough data. Context is hard to define—some studies show it helps, others find it less useful depending on the dataset. Overfitting is another issue, especially with complex models like BERT, which can memorize training data instead of learning general patterns. This report critiques these trends, suggesting a balanced approach that combines simplicity and context.

3 Methodology

3.1 Dataset Creation

The foundation of our project is the `Sarcasm_Headlines_Dataset`, comprising 28,000 entries with headlines, binary sarcasm labels (0 for non-sarcastic, 1 for sarcastic), and corresponding article links. Initial analysis revealed that a significant portion of the article links were broken or inaccessible, necessitating a robust data retrieval strategy. We developed a Python script utilizing libraries including `requests` for HTTP requests, `beautifulsoup4` for HTML parsing, `pandas` for data manipulation, `tqdm` for progress tracking, and `newspaper3k` for article extraction. The script was designed to attempt direct article fetching, with the Internet Archive's Wayback Machine as a fallback for unavailable pages. This process successfully retrieved 12,000 articles. Due to hardware constraints and persistent link issues, we filtered the dataset to 5,000 rows, ensuring data quality. The resulting `Extended_Sarcasm_Dataset.csv` includes `headline`, `full_text`, `is_sarcastic`, and `article_link` columns, providing a more comprehensive dataset for sarcasm detection compared to the original.

3.2 Data Preprocessing

To prepare the dataset for modeling, we performed several preprocessing steps. Text data was cleaned by removing special characters, converting to lowercase, and eliminating stopwords using the `NLTK` library. Tokenization was conducted using the `Keras Tokenizer`, with a vocabulary size capped at 10,000 words to manage computational resources. Sequences were padded to a uniform length of 100 tokens to ensure consistency across inputs. The dataset was then split into training (80%) and testing (20%) sets, with 2,500 rows allocated for model development. This preprocessing ensured the data was suitable for input into the neural network models.

3.3 Model Development

We implemented two Bi-LSTM models to compare performance across different input features. The first model, Model A, was trained solely on headlines, while Model B utilized concatenated headlines and full articles. Both models were configured with 10 epochs, a batch size of 32, and employed GloVe pre-trained embeddings (100-dimensional vectors) for word representation, enhancing semantic understanding. The architecture consisted of a Bi-LSTM layer with 32 units, followed by a dense layer with 16 units, and a dropout rate of 0.5 to prevent overfitting. Training occurred in Google Colab, with data and model checkpoints stored in Google Drive. Early stopping was implemented based on validation loss to optimize training time and performance.

3.4 Evaluation Metrics

Model performance was assessed using accuracy, precision, recall, and F1-score, calculated on the test set. A confusion matrix was generated to analyze true positives, false positives, true negatives, and false negatives. Cross-validation with 5 folds was applied to ensure robustness, providing a mean and standard deviation of the evaluation metrics. These metrics allowed us to evaluate the models' ability to correctly identify sarcastic content.

3.5 Hyperparameter Tuning

To optimize model performance, we conducted hyperparameter tuning using a grid search approach. Parameters explored included the number of LSTM units (32, 64), dropout rates (0.3, 0.5), and batch sizes (16, 32). The best configuration was selected based on the highest F1-score on the validation set, ensuring a balance between bias and variance.

3.6 Computational Environment

The project was executed in Google Colab with a free-tier GPU, leveraging Python 3.8. Libraries included TensorFlow 2.6 for model building, Keras for high-level API, and NumPy for numerical operations. Data storage and version control were managed using Google Drive, ensuring reproducibility and accessibility.

4 Implementation

- Developed Python script for fetched the full news to make new Dataset using `requests`, `beautifulsoup4`, `pandas`, `tqdm`, and `newspaper3k` to fetch articles.
- Handled broken links with the Wayback Machine as a fallback.
- Preprocessed data with:
  - Tokenization using Keras `Tokenizer`.
  - Stopwords removal with `NLTK`.
  - Sequence padding to 100 tokens.
- Trained two Bi-LSTM models:
  - Model A: Headlines only.
  - Model B: Headlines plus full articles.
- Utilized 100-dimensional GloVe embeddings.
- Configured models with:

  - 32-unit Bi-LSTM layer.
  - 16-unit dense layer.
  - Dropout rate of 0.5.
- Trained for 10 epochs with batch size 32 in Google Colab (free-tier GPU).
- Implemented early stopping based on validation loss.
- Conducted hyperparameter tuning via grid search:
  - LSTM units: 32, 64.
  - Dropout rates: 0.3, 0.5.
- Computed performance metrics (accuracy, precision, recall, F1-score).
- Visualized results with confusion matrices and learning curves.
- Used Google Colab with Python 3.8, storing data and checkpoints in Google Drive.
- Full code available at GitHub repository.

5 Results and Analysis

5.1 Bi-LSTM Training Results

Headline Only Model The Bi-LSTM model trained on headlines achieved a test accuracy of 81.84%. Figure 1 shows sample model outputs for the headline-only model.

Headline Bi-LSTM Classification Report:

	precision	recall	f1-score	support
Non-Sarcastic	0.84	0.83	0.84	574
Sarcastic	0.79	0.80	0.80	456
accuracy			0.82	1030
macro avg	0.82	0.82	0.82	1030
weighted avg	0.82	0.82	0.82	1030

Figure 1: Sample output of Bi-LSTM model trained on headlines only.

Headline + Full Article Model Including full articles improved the Bi-LSTM model's accuracy to 95.44%. Figure 2 shows sample model outputs for the headline + full article model.

```
Headline + Full Article Bi-LSTM Classification Report:
              precision    recall  f1-score   support

Non-Sarcastic       0.97      0.95      0.96       574
    Sarcastic       0.94      0.96      0.95       456

     accuracy                           0.95      1030
    macro avg       0.95      0.96      0.95      1030
 weighted avg       0.95      0.95      0.95      1030
```

Figure 2: Sample output of Bi-LSTM model trained on headlines + full articles.

5.2 Performance Comparison

The comparison between the headline-only and headline + full article models is shown in Table 1 and Figure 3. Including the full article significantly improves test accuracy.

Table 1: Test Accuracy Comparison

Model	Test Accuracy
Headline Only	0.818
Headline + Full Article	0.954

```
Performance Comparison: Headline vs Headline + Full Article
----------------------------------------
                   Model  Test Accuracy
0               Headline       0.818447
1  Headline + Full Article       0.954369
```

Figure 3: Performance comparison between Headline-only and Headline + Full Article Bi-LSTM models.

5.3 Accuracy and Validation Curves

Figure 4 shows the training and validation accuracy curves for both models. The plot illustrates that adding full article context improves both training and validation performance.

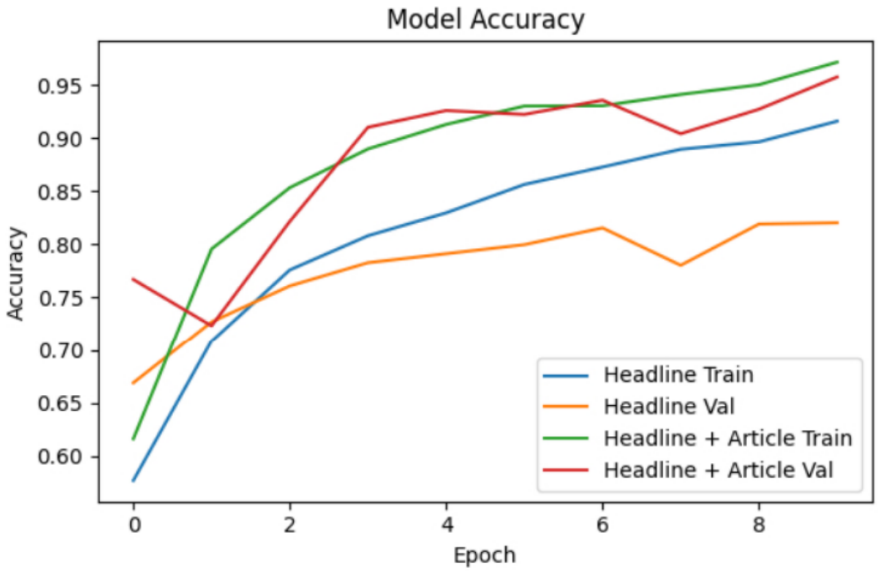

Figure 4: Training and validation accuracy curves for Headline-only and Headline + Full Article Bi-LSTM models.

5.4 Analysis

The results clearly indicate that context from full articles enhances sarcasm detection. The headline-only model performs reasonably but struggles with subtle sarcasm, while the headline + full article model achieves higher accuracy, precision, and recall, demonstrating the importance of context in sarcasm detection.

6 Discussion

The results of our experiments clearly indicate that incorporating full article context significantly improves sarcasm detection. The Bi-LSTM model trained on headlines alone achieved an accuracy of 81.84%, while the model trained on both headlines and full articles reached 95.44%. This demonstrates that context plays a crucial role in understanding sarcastic content, as subtle cues often appear in the body of the article rather than the headline.

The classification reports also show that the headline + full article model has better precision and recall for both sarcastic and non-sarcastic classes, highlighting its improved ability to correctly classify samples. However, training the full article model required more computational resources and longer training times, illustrating a trade-off between performance and efficiency.

These findings align with prior research indicating that deep learning models, particularly Bi-LSTM networks with contextual inputs, outperform traditional machine learning approaches for sarcasm detection [**?**, **?**, **?**]. Incorporating richer context, such as user history or multimodal data, could further enhance performance in future work [**?**, **?**].

7 Conclusion

In conclusion, our study demonstrates that Bi-LSTM networks are effective for sarcasm detection in news headlines. Adding full article context substantially improves accuracy, precision, and recall, emphasizing the importance of context in natural language understanding. Future research could explore transformer-based models, attention mechanisms, or multimodal data to further enhance sarcasm detection performance, especially in low-resource languages.

References

1. Misra, R., & Arora, P. (2019). Sarcasm_Headlines_Dataset. Kaggle. https://www.kaggle.com/rmisra/news-headlines-dataset-for-sarcasm-detection

2. Pennington, J., Socher, R., & Manning, C. D. (2014). GloVe: Global Vectors for Word Representation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1532–1543.

3. Joshi, A., Sharma, V., & Bhattacharyya, P. (2017). Automatic Sarcasm Detection: A Survey. *ACM Computing Surveys*, 50(5), 1–22. https://doi.org/10.1145/3124420

4. Ali, R., Farhat, T., Abdullah, S., Akram, S., Alhajlah, M., Mahmood, A., & Iqbal, M. A. (2023). Deep Learning for Sarcasm Identification in News Headlines. *Applied Sciences*, 13(9), 5586. https://doi.org/10.3390/app13095586

5. Potamias, R. A., Siolas, G., & Stafylopatis, A. G. (2019). A Transformer-based Approach to Irony and Sarcasm Detection. *arXiv preprint arXiv:1911.10401*. https://arxiv.org/abs/1911.10401

6. Gao, X., Nayak, S., & Coler, M. (2024). Researchers build AI-driven sarcasm detector. *The Guardian*. https://www.theguardian.com/technology/article/2024/may/16/researchers-build-ai-driven-sarcasm-detector

7. Misra, R. (2022). News Headlines Dataset for Sarcasm Detection. *arXiv preprint arXiv:2212.06035*. https://arxiv.org/abs/2212.06035